11. Berkeley Database API

Created: March 22, 2004
Updated: March 22, 2004

Overview

The overview for this chapter consists of the following topics:

- Introduction
- Chapter Outline

Introduction

BDB library is part of NCBI C++ Toolkit and serves as a high level interface to Berkeley DB. The primary purpose of the library is to provide tools for work with flat file, federated databases. BDB library incorporates number of Berkeley DB services and for detail understanding of how it works it's good to study the original Berkeley DB documentation form http://www.sleepycat.com [http://www.sleepycat.com]. BDB library is compatible with Berkeley DB v. 4.1 and higher. BDB library as it is right now is architecturally different from dbapi library and does not follow its design. BDB is intended for use by software developers who need small footprint high-performance database capabilities with zero administration. Database in this case becomes tightly integrated with the application code.

Chapter Outline

The following is an outline of the topics presented in this chapter:

- Major features of the BDB library

Major features of the BDB library

BDB library intention is to provide tools for development of specialized data storages in applications not having access to centralized RDBMS.

- C++ wrapper on top of Berkeley DB. BDB library takes care of many of ultra low level details exposed to C programmer by Berkeley DB. BDB implements B-Tree file access (both keyed and sequential), environments, cursors, transactions.
- Error checking. All error codes coming from Berkeley DB are analyzed and processed in a manner common to all other components of C++ Toolkit. When an error situation is detected, BDB library sends an exception which is reported by the diagnostic services and can be handled by the calling application like any other toolkit exception.

- Support for relational table structure and different data types. Berkeley DB itself is 'type agnostic' and provides no means to manipulate data types. But for many cases clear data type support can save a lot of work. Toolkit implements all major scalar data types so it can be used like a regular database.
- Cross platform compatibility. BDB databases can be transferred across platforms without reconverting the data. BDB tracks the fact that the database was created as big-endian or little-endian and does the conversion transparently when the database migrates.
- Easy BLOBs. BDB library supports keyed BLOB storages. BLOBs can be streamed to and from the database. A set of additional interfaces has been written to simplify the BLOB access in comparison to the original Berkeley DB C library.
- Disk based cache interface. BDB library implements cache disk cache service used by other toolkit components to minimize client-server traffic and store parts of the data locally. Different cache management and data expiration policies have been put in place.
- Database maps. BDB library includes templated classes similar to STL map and multimap but persistently stories the map content in Berkeley DB files.